Universidad Francisco de Paula Santander

Ocaña - Colombia

Vigilada Mineducación

# Docente:

## WILDER ANDRÉS DUARTE NEIRA

### Ingeniero de Sistemas

### Esp. en Auditoría de Sistemas

waduarten@ufpso.edu.co / 316-866-7751

# MARATÓN

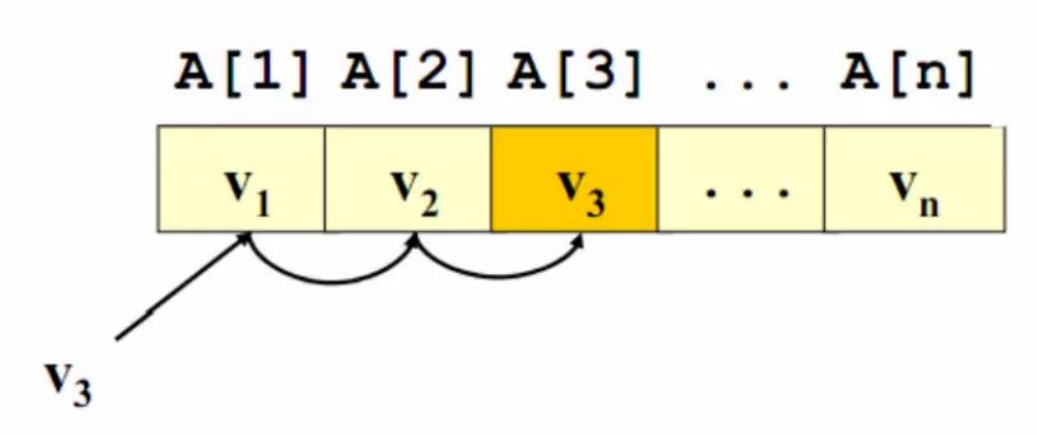
# BÚSQUEDA LINEAL

También llamada búsqueda secuencial. Es la más intuitiva que existe.

Este algoritmo compara uno a uno los elementos del arreglo indicando si el número buscado existe, hasta encontrarlo o recorrerlo por completo. En éste caso no se ordenará la lista de elementos.

En este caso es importante conocer la cantidad de elementos donde voy hacer la búsqueda.

# BÚSQUEDA LINEAL



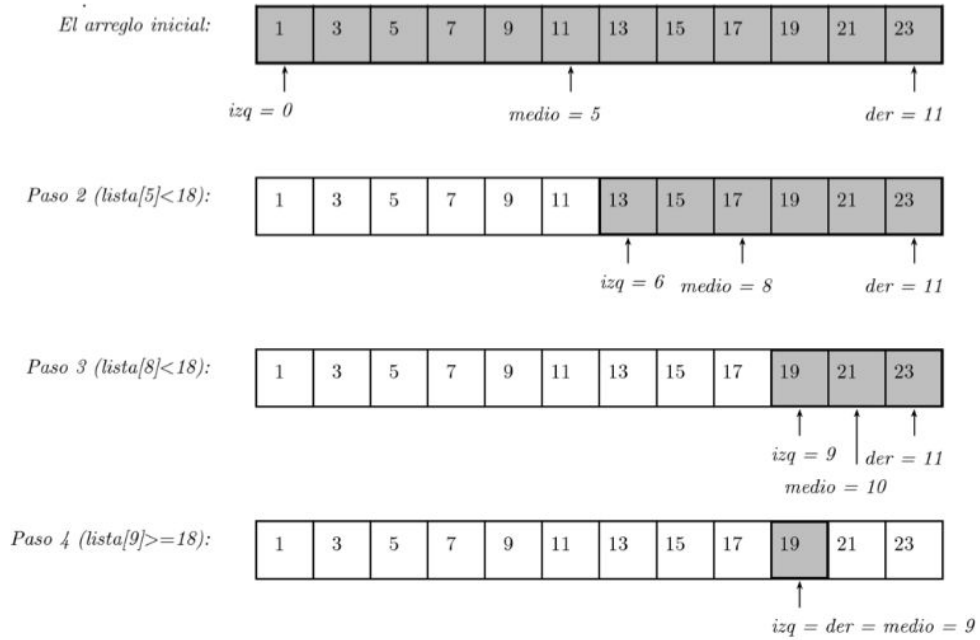
# BÚSQUEDA BINARIA

Este algoritmo permite buscar de una manera más eficiente un dato dentro de un arreglo, para hacer esto se determina el elemento central del arreglo y se compara con el valor que se está buscando, si coincide termina la búsqueda y en caso de no ser así se determina si el dato es mayor o menor que el elemento central, de esta forma se elimina una mitad del arreglo junto con el elemento central para repetir el proceso hasta encontrarlo o tener solo un elemento en el arreglo.

Para poder aplicar este algoritmo se requiere que el arreglo este ordenado. Esta basado en el paradigma divide y vencerás.

# BÚSQUEDA BINARIA

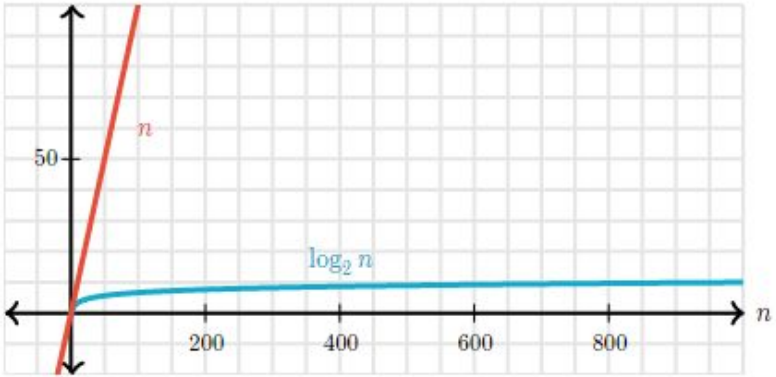
En el gráfico que se incluye a continuación, vemos qué pasa cuando se busca el valor `18` en la lista `[1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23]`.



# BÚSQUEDA LINEAL VS BÚSQUEDA BINARIA

| $n$ | $\log_2 n$ |
|---|---|
| 1 | 0 |
| 2 | 1 |
| 4 | 2 |
| 8 | 3 |
| 16 | 4 |
| 32 | 5 |
| 64 | 6 |
| 128 | 7 |
| 256 | 8 |
| 512 | 9 |
| 1024 | 10 |
| 1,048,576 | 20 |
| 2,097,152 | 21 |

Compara $n$ contra $\log_2 n$ a continuación:



# BÚSQUEDA LINEAL

```
01  function linearSearch(value, list) {
02      let found = false;
03      let position = -1;
04      let index = 0;
05
06      while(!found && index < list.length) {
07          if(list[index] == value) {
08              found = true;
09              position = index;
10          } else {
11              index += 1;
12          }
13      }
14      return position;
15  }
```

# BÚSQUEDA BINARIA

```
01  function binarySearch(value, list) {
02      let first = 0;    //left endpoint
03      let last = list.length - 1;   //right endpoint
04      let position = -1;
05      let found = false;
06      let middle;
07
08      while (found === false && first <= last) {
09          middle = Math.floor((first + last)/2);
10          if (list[middle] == value) {
11              found = true;
12              position = middle;
13          } else if (list[middle] > value) {  //if in lower half
14              last = middle - 1;
15          } else {  //in in upper half
16              first = middle + 1;
17          }
18      }
19      return position;
20  }
```